Consider Multiple Solutions

Jeff Patton

I recently listened to criticism of agile development from Bill Buxton, a leading member of the user-centered-design community. He suggested that practitioners of agile development lock onto a software solution early, then use iteration to refine it. To paraphrase Bill, he said that while agile's short, time-boxed development iterations let practitioners evaluate what they've built and then make changes and improvements, they seldom consider exploring multiple possible solutions or throwing the whole thing out and starting over. (Learn more about Buxton's ideas in *Sketching User Experiences*, Morgan Kaufmann, 2007.)

After feeling a little indignant that my chosen process approach had an alleged flaw, I realized that Bill was mostly right. In the dozens of agile projects I'd participated in and observed, I couldn't think of a regular practice used by any team to consider multiple solution ideas before committing to one. We considered alternative ideas only when we didn't like the one we were currently using. "If it ain't broke, don't fix it." We hadn't considered that there could be an even better idea around the corner.

It makes sense that if we consider many possible solutions, we'll likely arrive at something better than the first one we come up with. But, I'll come clean here: I'm lazy. I tend to fall in love with my ideas pretty quickly. My attempts to come up with alternative solutions are half-hearted at best—especially if I've come up with a solution I and others feel good about.

While considering this dilemma, two good practice approaches landed in my lap. And, as it turned out, the solutions worked best for me when combined—which is probably another example of multiple ideas being better than one.

Before I go on, I need to clarify what I mean by *design*: it's the choices we make when we decide what to build. Basically, given some business goal or problem and some understanding of users and their current goals and practices, what software would we build that best meets all these goals? Given that solution choice (or the product of that design activity), we're left with engineering work (or the technical design activity) to make this solution a reality. Both types of design, figuring out what to build and figuring out how to build it, are important. Also, they're often conjoined in a way that makes it hard to tell where one ends and the other begins. For this column's purposes, when I use the words "solution" or "design," I'm thinking about the product we choose to build and how it looks and behaves when used.

Sketchboarding

I'm impressed with the smart people at Adaptive Path who continue to come up with great ideas to make design quick and collaborative. Sketchboarding is a recent good idea from that company.

Participants in a sketchboarding exercise start with a blank roll of butcher paper or craft wrapping paper, usually available in the packaging supplies section of an office supply store. A team member rolls out and cuts off 10 feet or so and tapes it to the wall. This is the foundation of a sketchboard.

If we were participating in a skethboarding exercise, we'd start by taping to the paper what we know about the problem we're solving. Goals of the business, details about the current application's users, diagrams of their workflow, screenshots of their current tools annotated with problem areas, quotes from unhappy users and stakeholders—anything that would help us get a sense of

the problems we're trying to solve with our software.

From there, we'd begin to sketch solutions on separate sheets of paper and affix them to the sketchboard with tape. Solutions are sketched as little thumbnails of possible computer screens, sketched close-ups of possible user interfaces, sticky notes with good ideas written on them, or anything that helps communicate our brainstormed ideas.

Given the problem space and lots of lo-fi solution ideas in close proximity, we can than discuss with the stakeholders, users, and others both the problem and solution space and eventually arrive at a best solution that combines and builds on elements of the best brainstormed solutions. (More information about sketchboarding is at www.adaptivepath.com/ideas/essays/archives/000863.php.)

When I first heard about this concept, I was excited—but not excited enough to put it into practice. I didn't have ready access to the collaborators I wanted to brainstorm with—to bounce ideas off of. And, I still had my lazy problem to deal with—my lack of enthusiasm for coming up with more ideas when I felt like I already had a decent one.

Design studio

Smart interaction designers Jim Ungar and Jeff White of Jewelry Television (www.jewelrytelevision.com) came up with what they call the design studio approach. They wanted to get the developers and developer teams in their organization involved in making solution design choices. Why have all the fun themselves?

Actually, they weren't exactly having all the fun. In fact, it's common for most people in a design role to research their users, their goals, and current practices; brainstorm possible solutions, choose the best solution, then refine a user interface design—only to have it questioned or second-guessed by others who hadn't done the research or considered other options. When it comes to user interface design, everyone's a critic. Jeff and Jim wanted a way others could better understand what goes into good interaction design. They were also pressed for time, so they could use a little help. And finally, they worked in an agile development environment, which defers much of design activity till late in the pro-

cess and prides itself on collaboration. An ideal process solution would be quick, collaborative, save them time, and earn them the empathy they desired.

Jeff White thought back to the approach used in the design school he attended and decided to borrow it—and that led to the design studio approach.

Jim and Jeff gave their team members the material they needed to understand the problem space: the research they'd done on users, workflow models, information about problems and pain points. They then asked their team members to sketch potential user interface design solutions independently. Team members were asked to spend a short time sketching, no more than two hours.

Jim and Jeff then scheduled a one-day design studio workshop for all team members and asked everyone to bring in their design solutions. At the workshop, they started by reviewing the problem material. Then they reviewed each participant's solution idea with all the participants in the workshop. Everyone was able to say what they thought could be problematic about the solution and, more importantly, what was valuable about the solution idea. As the team discussed solution ideas, a recorder noted on flipcharts all the good ideas that could be included in a prospective solution along with the things to avoid. Then the team went to work to choose the best ideas and collaboratively design a best solution.

Combining design studio and sketchboarding

Recently, I had the opportunity to conduct a workshop with a group of smart business analysts. I suggested trying a combined version of both these practices, and the group agreed to go along with me. During the workshop, the whole group reviewed the existing problem space for a difficult project the company was working on. After discussing the problem and pasting up screenshots and other details, the team of analysts went to work brainstorming ideas. They didn't brainstorm in a big, loud discussion, rather they grabbed sheets of the templates generously supplied by Adaptive Path and began sketching ideas. We all agreed to time-box this sketching activity at 30 minutes.

After sketching solutions and taking a short break, we came back to discuss each solution. One at a time the solution designers, the business analysts in this case, came

up and posted their solutions on the long sheet of butcher paper. They then described their solution to the rest of the group, while all the participants made notes of what they thought were the best ideas in the solution. We recorded these ideas on sticky notes and fixed them to the solution on the sketchboard.

Before too long, we began to notice patterns of similar solutions. Those patterns of similar solutions gave participants ideas of even better solutions no one in the group of 10+ analysts had yet thought of.

At the end of this process—which took less than two hours end to end—we'd arrived with more great ideas in a shorter time than I'd ever seen before. What's more, everyone in the room became very familiar with the business problem we were trying to solve—the same experience Jim and Jeff had reported at Jewelry Television when conducting a design studio. We'd used the single sheet of paper to contain the problem information and solution ideas, just like the sketchboarding approach—but used the independently sketched ideas and review process described by Adaptive Path. Everyone in the room had good ideas that were unique or variations of others' good ideas. The resulting pool of ideas was richer than the single business analyst had been able to come up with when working with stakeholders prior to this workshop. That original analyst that brought in the tough problem happily rolled up the sketchboard to take back to his stakeholders.

These two techniques, sketchboarding and design studio, combined easily to create a quick, collaborative design practice that let us quickly ideate over many solutions. In addition to having fun with the approach, I left the exercise really understanding the value of ideating over multiple solutions before getting lost in the implementation details of any one.

Bill Buxton was right that we often don't consider enough possibilities before moving into implementation. But, although he might be right about some teams, there's certainly evidence that others are inventing and adopting new practice to consider more ideas. I'm happy Bill harped on a point I'd neglected and gave me the motivation to try a new approach. ⬚

Jeff Patton is an independent consultant. Contact him at jpatton@acm.org.